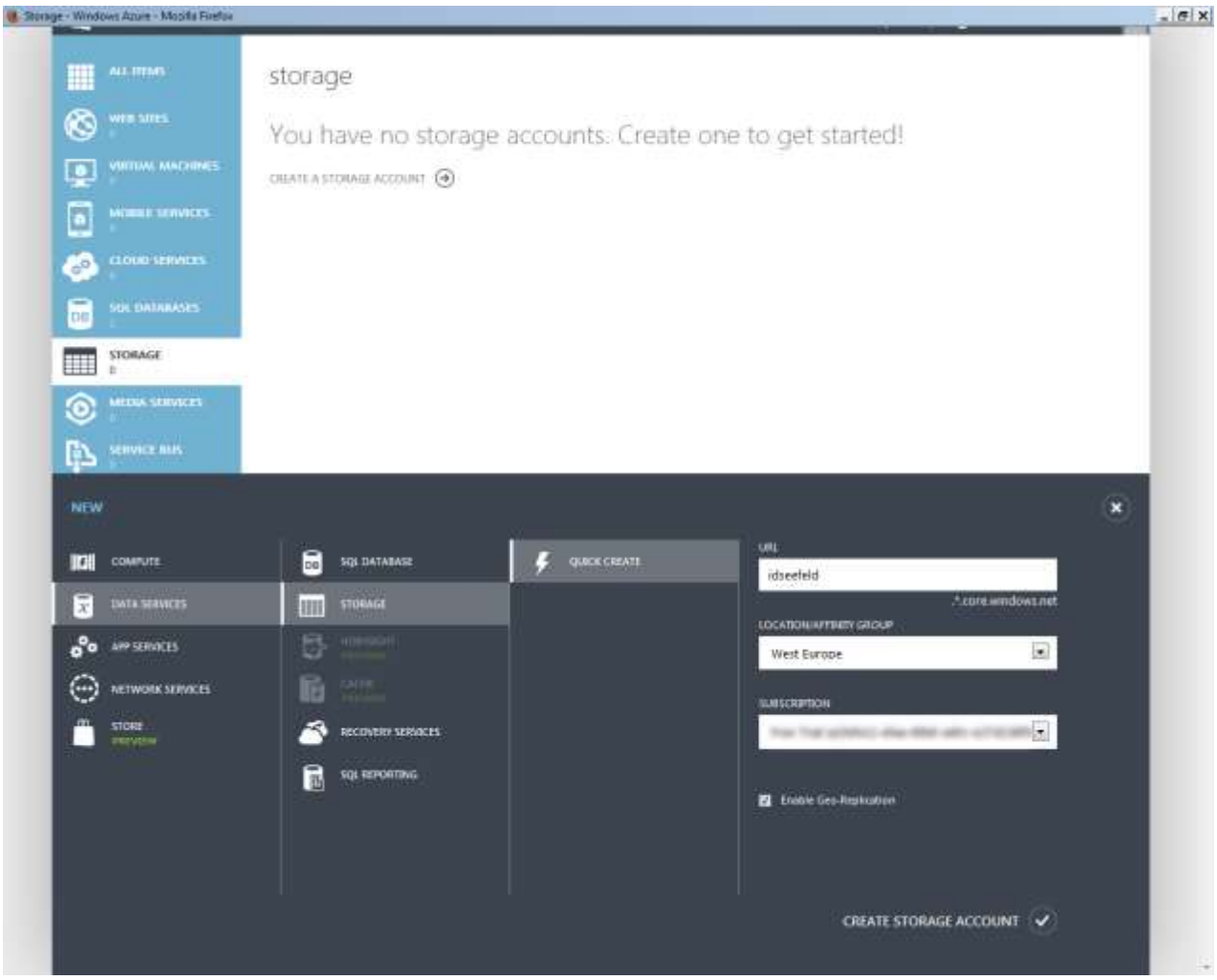
Azure Blob Storage Provider Documentation (Version 1.0.10)

What is the Azure Blob Storage Provider for Umbraco?

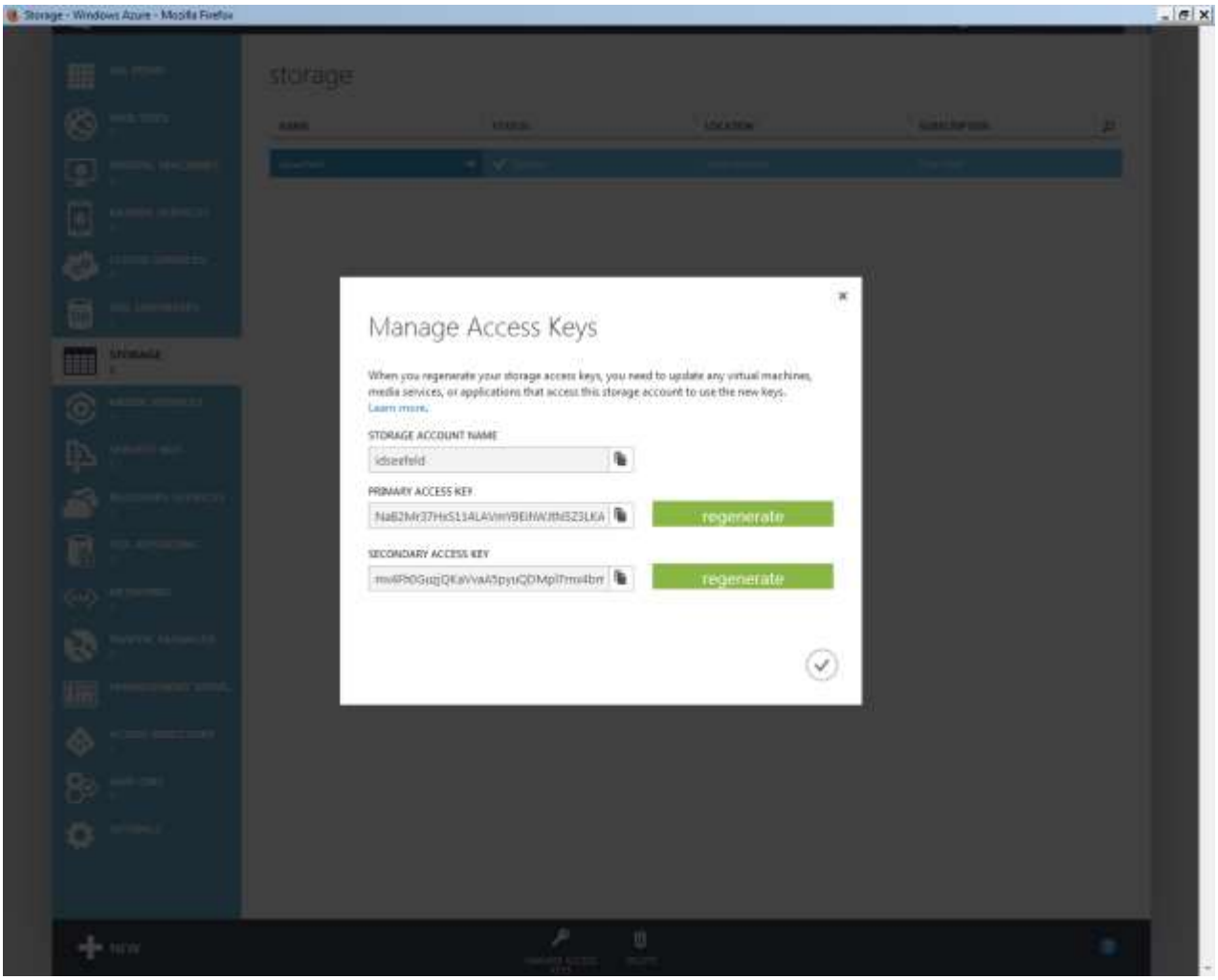
The Azure Blob Storage Provider replaces Umbracos default provider for media files. ***Full trust is required!***
The following steps show how to setup the *ABSP*:

1. At first you create the Azure storage.

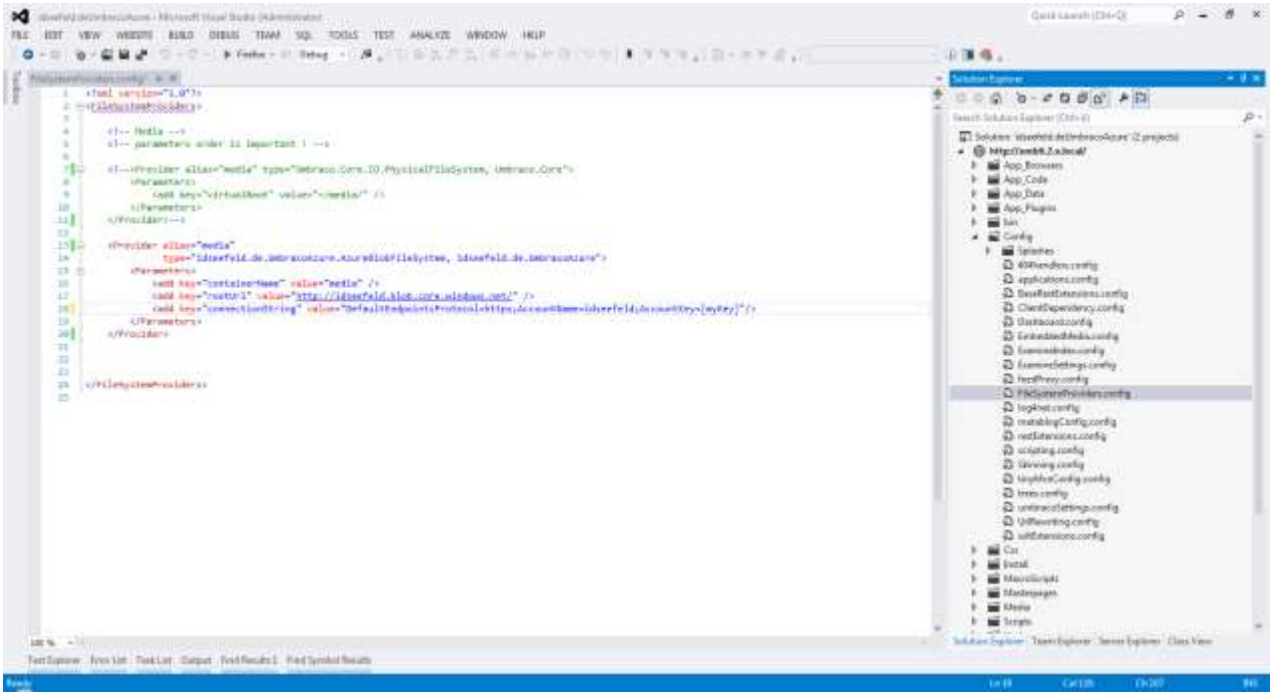


The Result

2. Copy one of the two generated keys.



3. Open file ~/Config/FileSystemProviders.config of your Umbraco installation and paste your account name and key into the configuration:



```
<?xml version="1.0"?>
<FileSystemProviders>
    <!-- Media -->
    <!-- parameters order is important ! -->

    <Provider alias="media"
                          type="idseefeld.de.UmbracoAzure.AzureBlobFileSystem,
idseefeld.de.UmbracoAzure">
                <Parameters>
                        <add key="containerName" value="media" />
                        <add key="rootUrl" value="http://[myAccountName].blob.core.windows.net/" />
                        <add key="connectionString"
value="DefaultEndpointsProtocol=https;AccountName=[myAccountName];AccountKey=[myAccountKey]"/>
                        <add key="mimetypes" value="" />
                        <add key="cacheControl" value="*|public, max-age=31536000;js|no-store" />
                </Parameters>
        </Provider>

</FileSystemProviders>
```

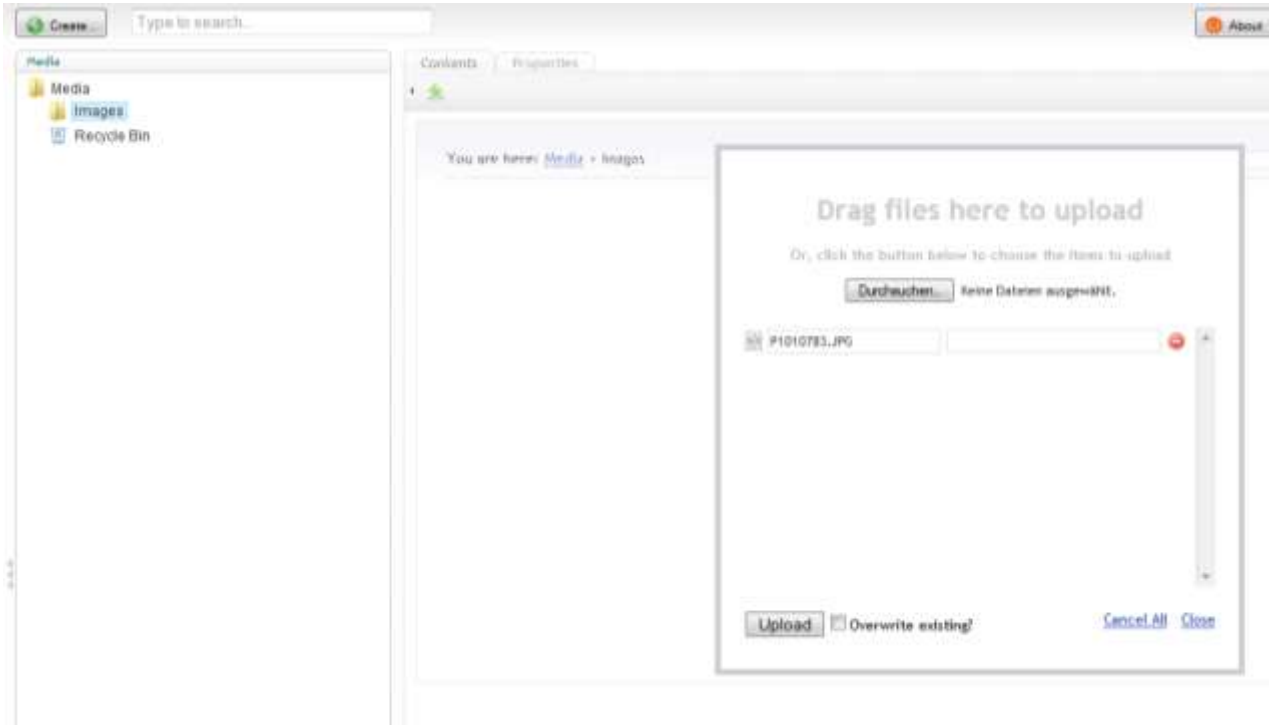
You can configure Cache-control for several file types. Add your settings to the parameters with key=”cacheControl”. Start with the wildcard (*) setting and separate the file type extension by | (pipe) from the cache-control settings. Separate each **file type | cache-control** group by ; (semicolon).
Allowed cache-controls are:
`public` *or* `public, max-age=12345678` (*or simply:* `max-age=12345678`)
`private` *or* `private, max-age=12345678`
`no-store`
`no-cache`
The max-age value is defined in seconds - i.e. 31536000 means one year (60 x 60 x 24 x 365).
For backward compatibility you have to add mimetypes and cacheControl keys. However, you can leave the value of mimetypes blank.

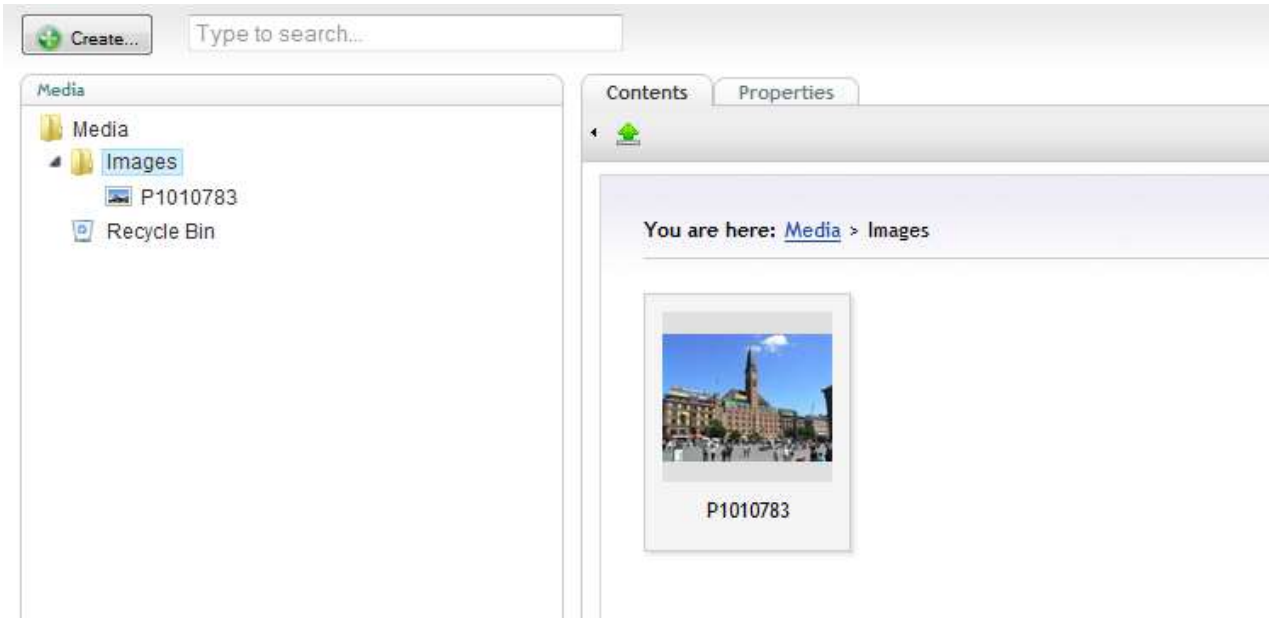
Since version 1.0.10.4 you can add a setting to the appSettings section of your Umbraco installation’s web.config:

```
<add key="AzureBlobFileSystem.UseWAStorageEmulator" value="true" />
```

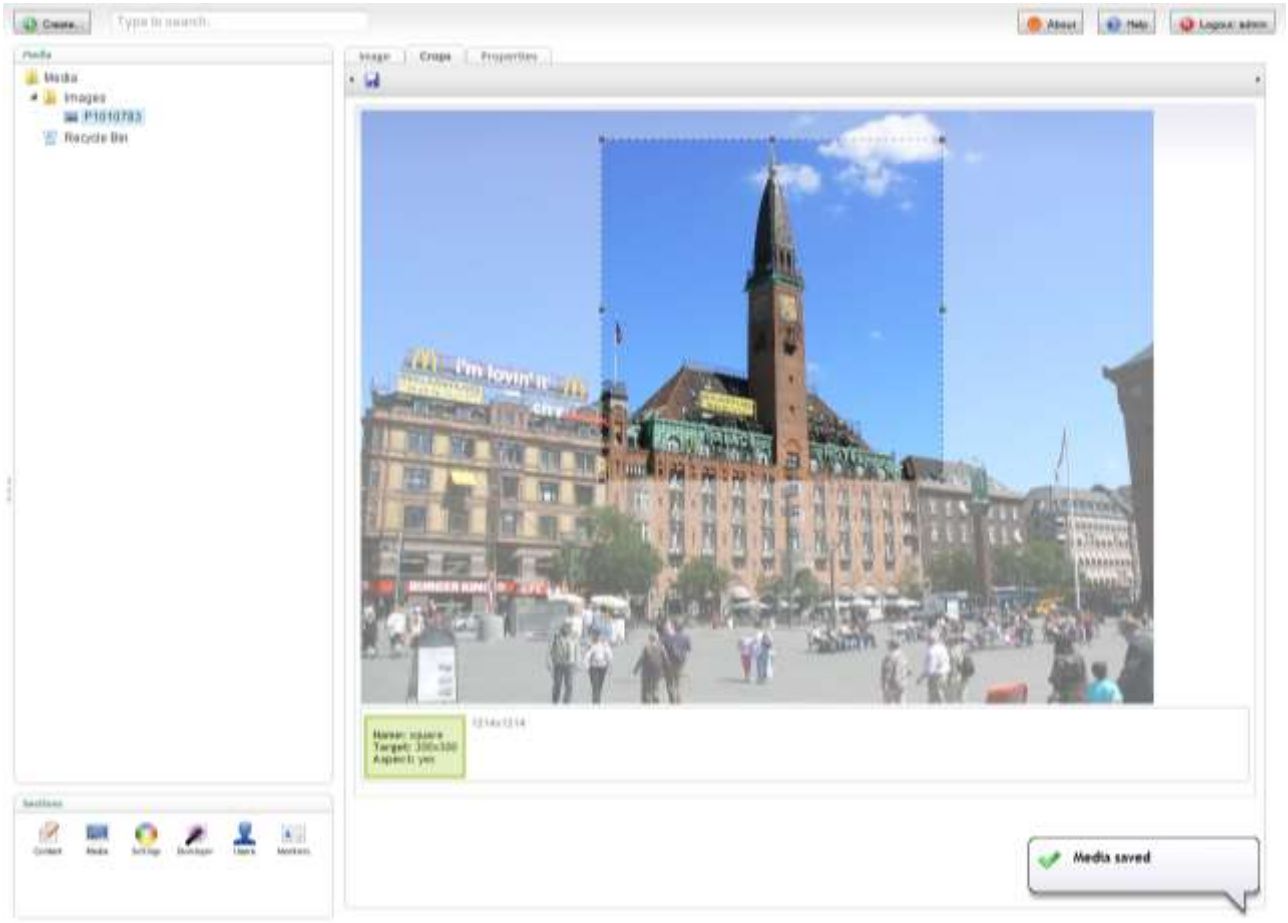
When AzureBlobFileSystem.UseWAStorageEmulator is true, rootUrl and connectionString are set to the Windows Azure Development Storage Emulator. The emulator is part of the Windows Azure SDK.

4. Restart the web server (e. g. touch web.config) and upload an image.

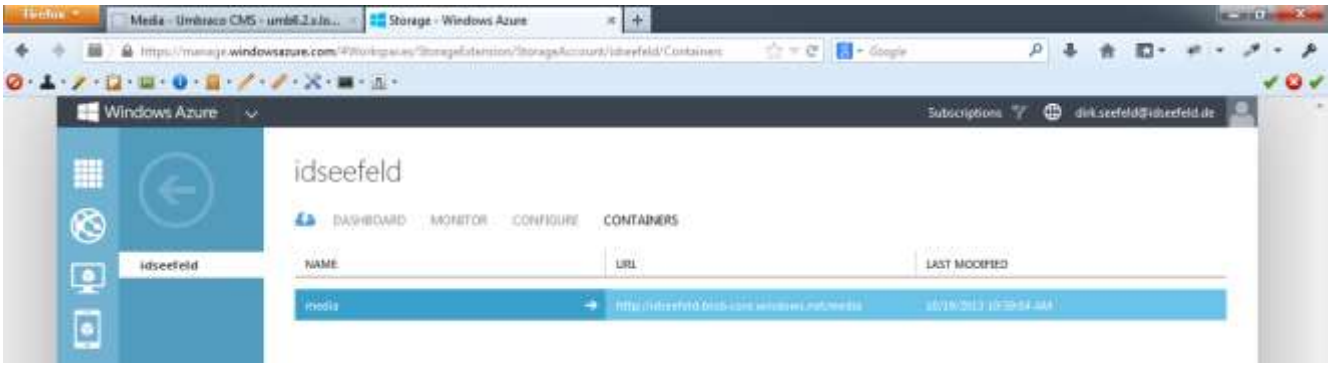




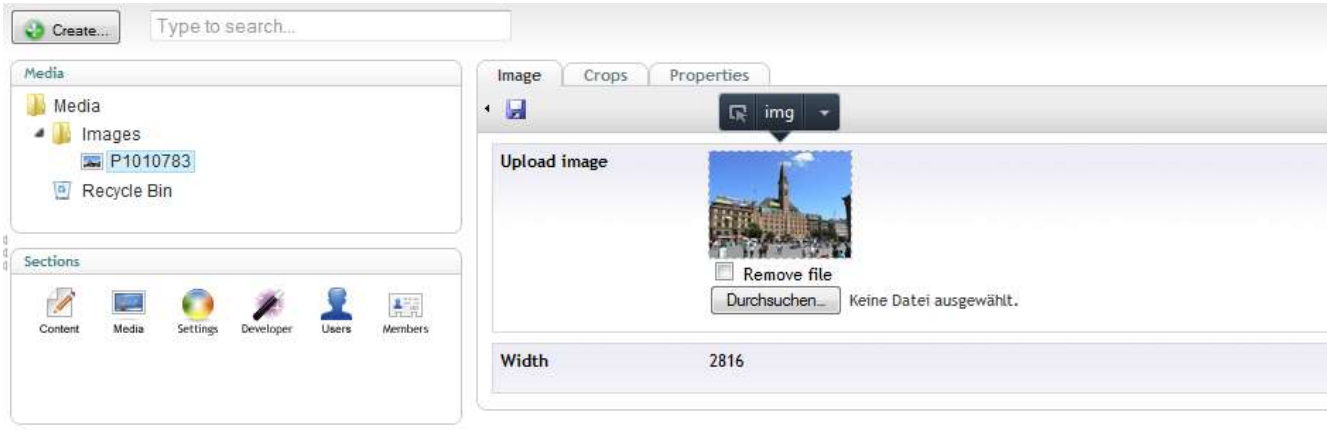
In my example I have previously installed the Image Cropper Extended and defined crops for media and content (for details see Image Cropper Extended documentation). *Just to show that it works!*

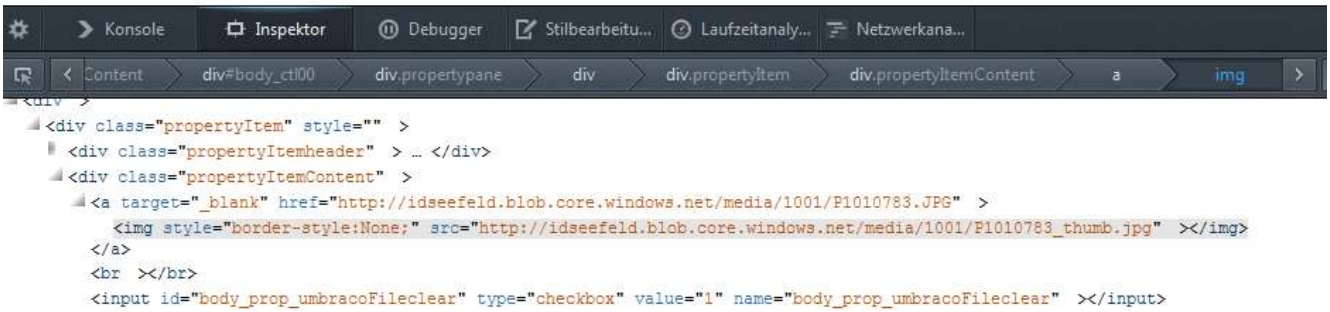


5. Check that your uploaded images are in the cloud ...

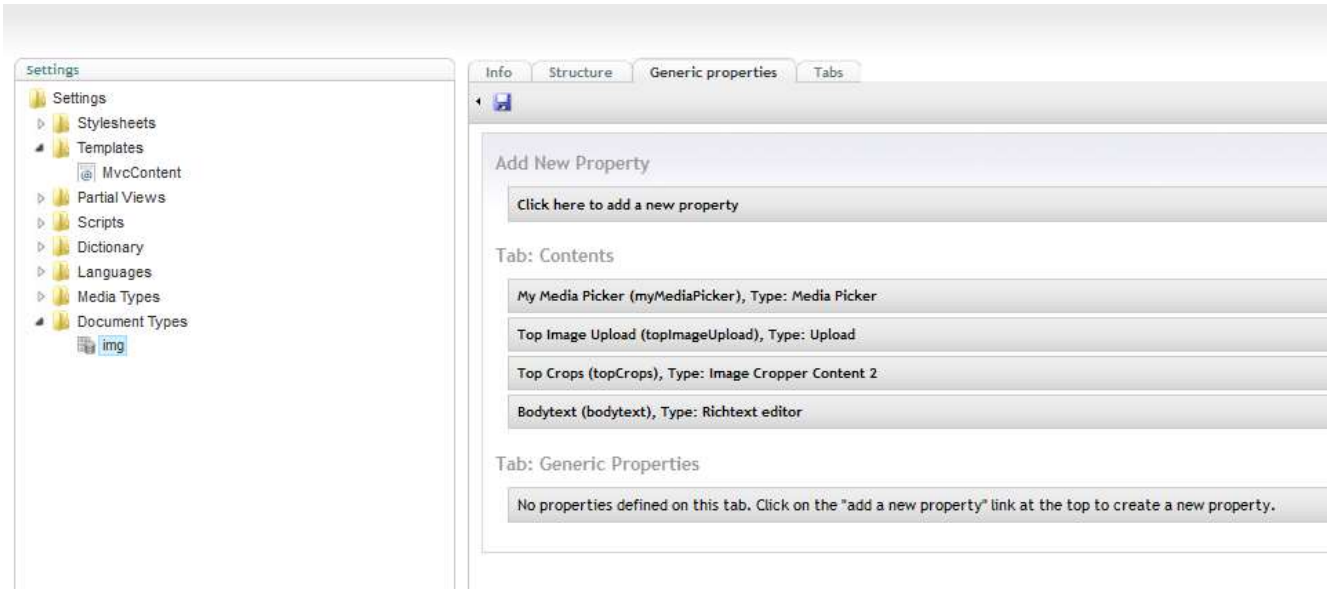


6. … or explore the html.

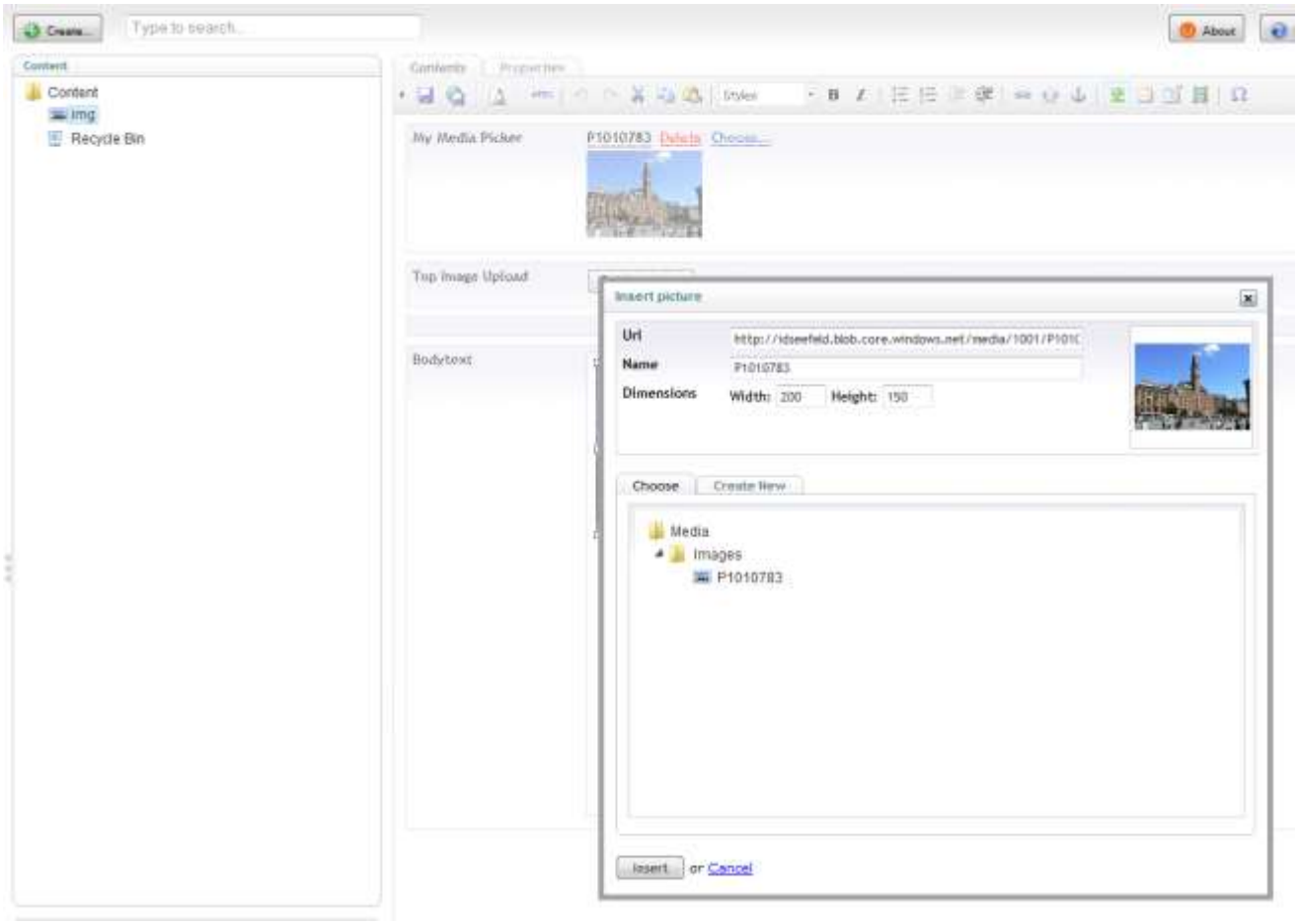


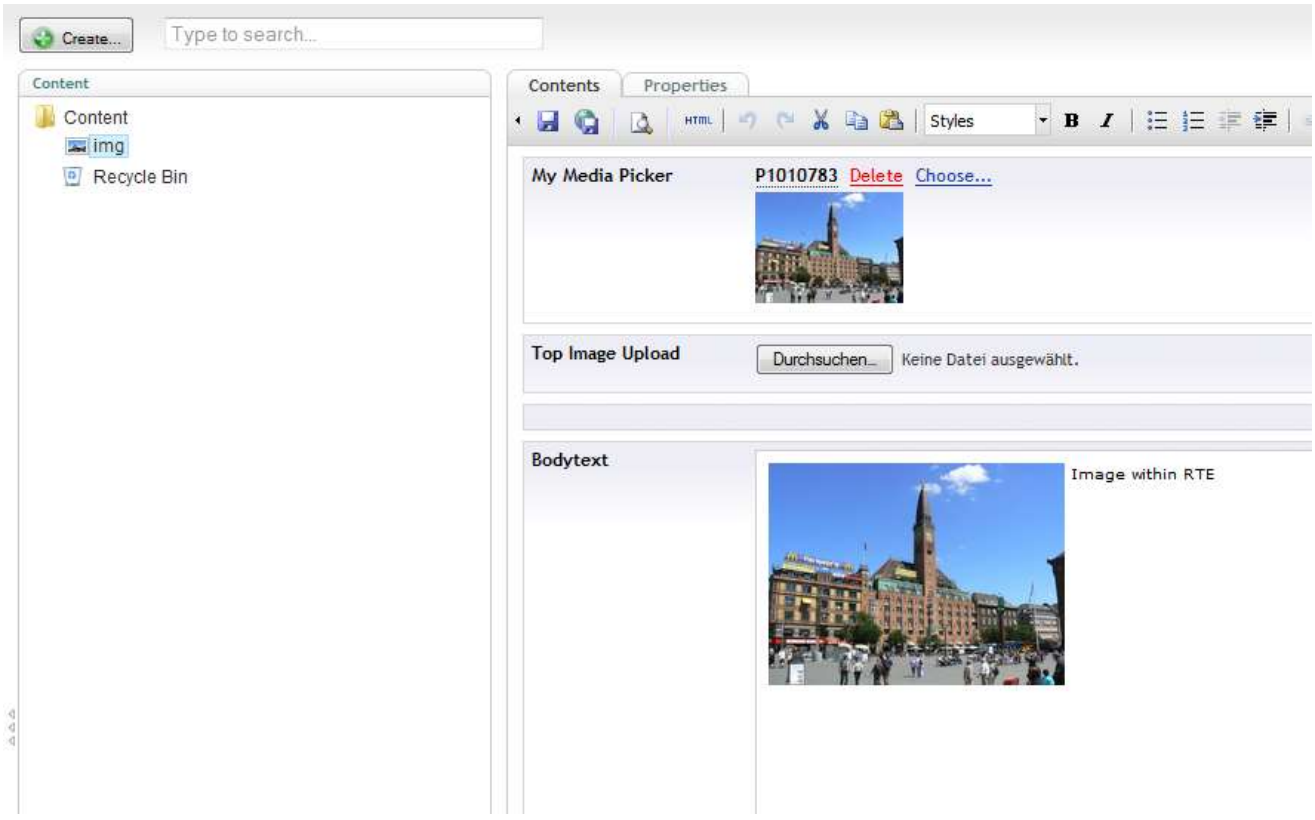


7. Example of a document type

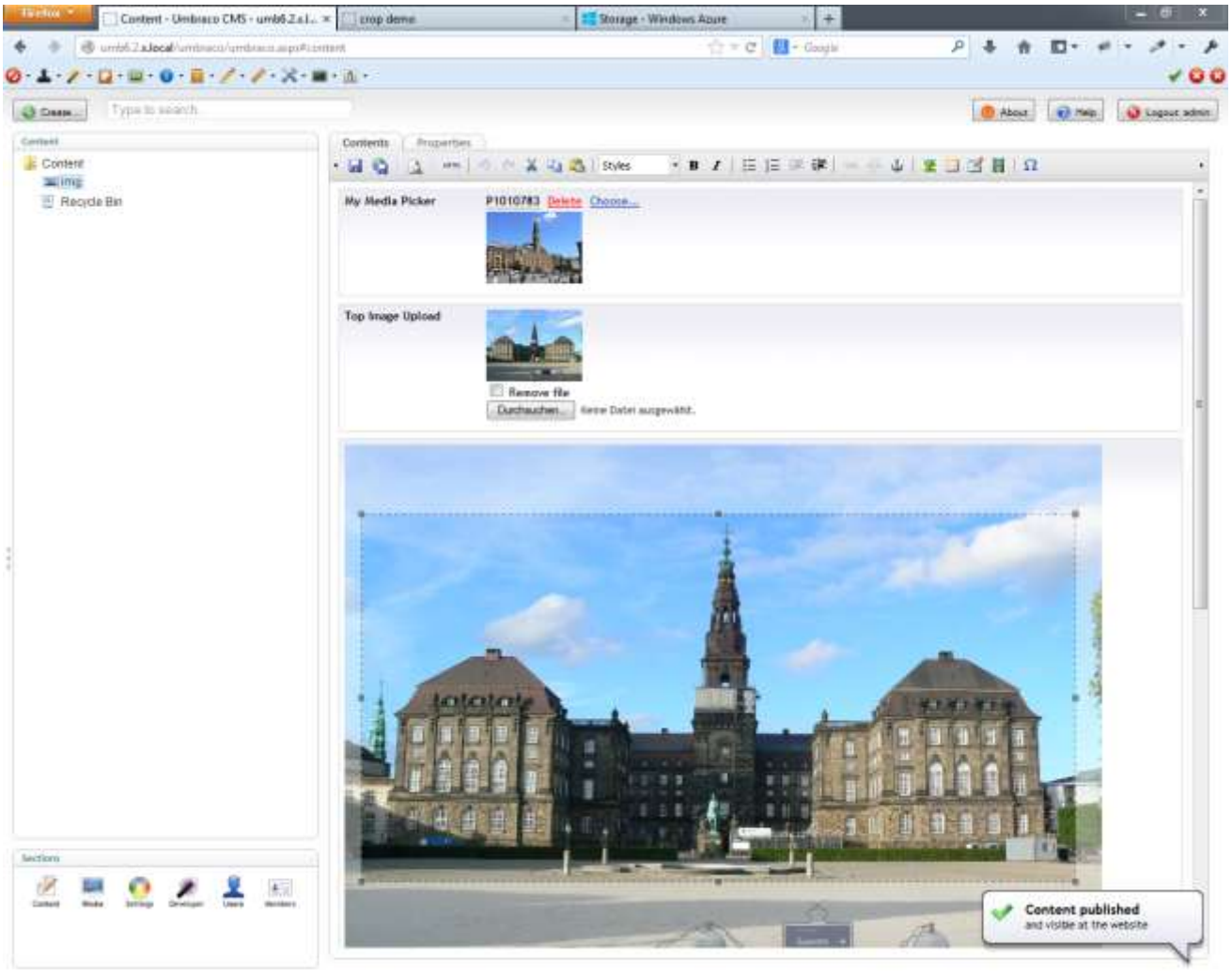


8. Create a node of the *img* document type and choose an image with the media Picker and insert one into the rich text editor.

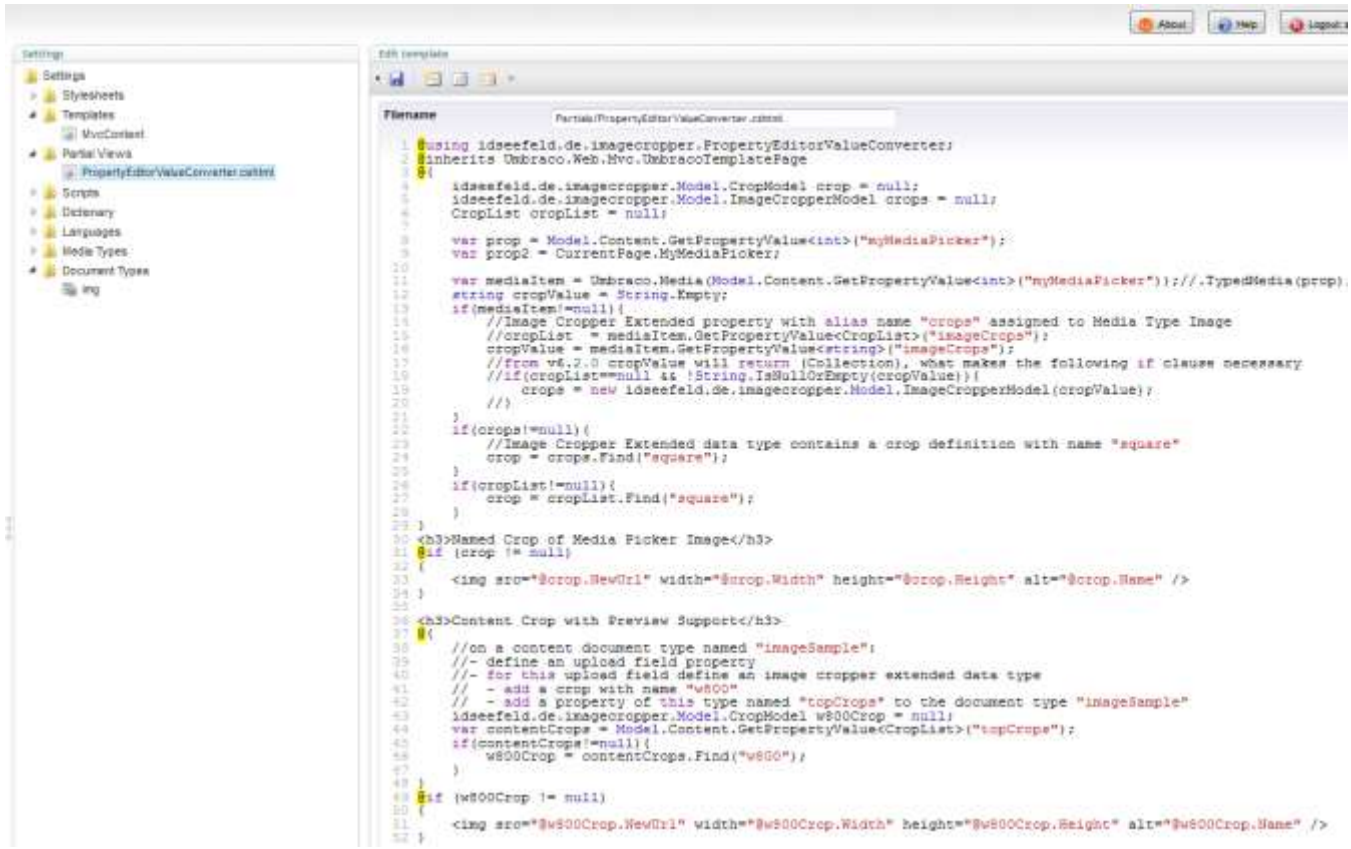




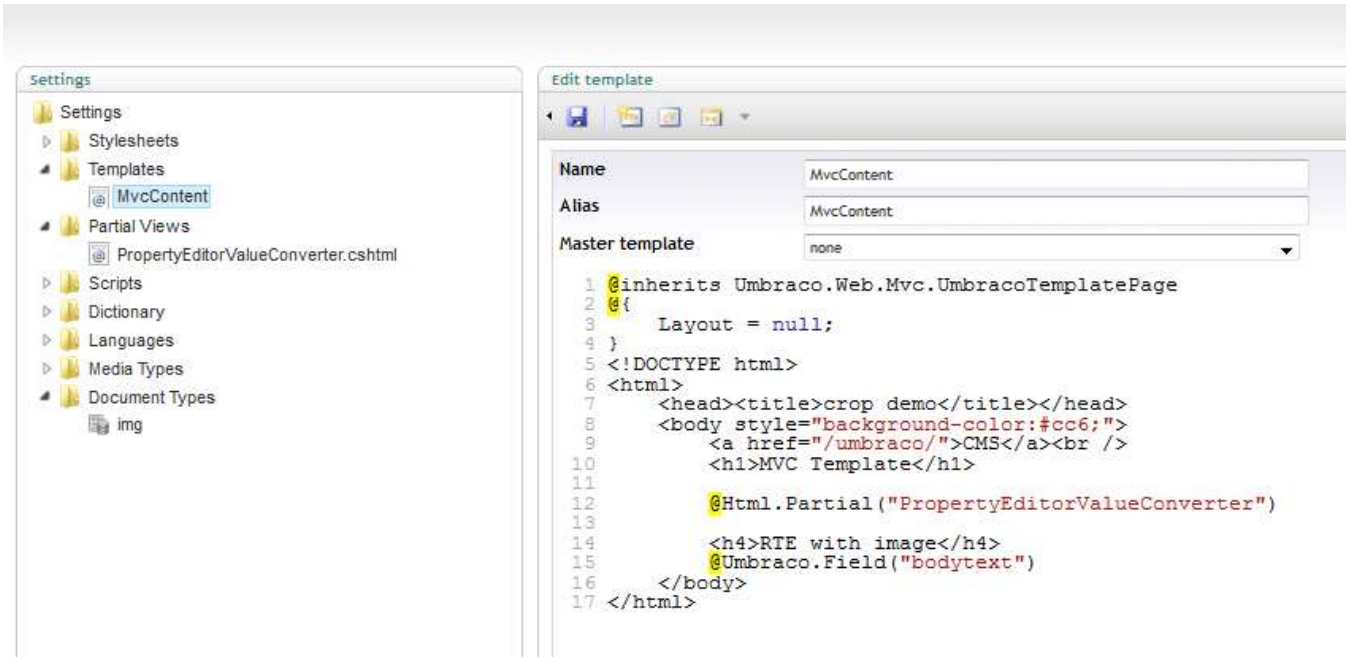
9. Upload another image with the upload, adjust the associated cropper and save & publish the node.



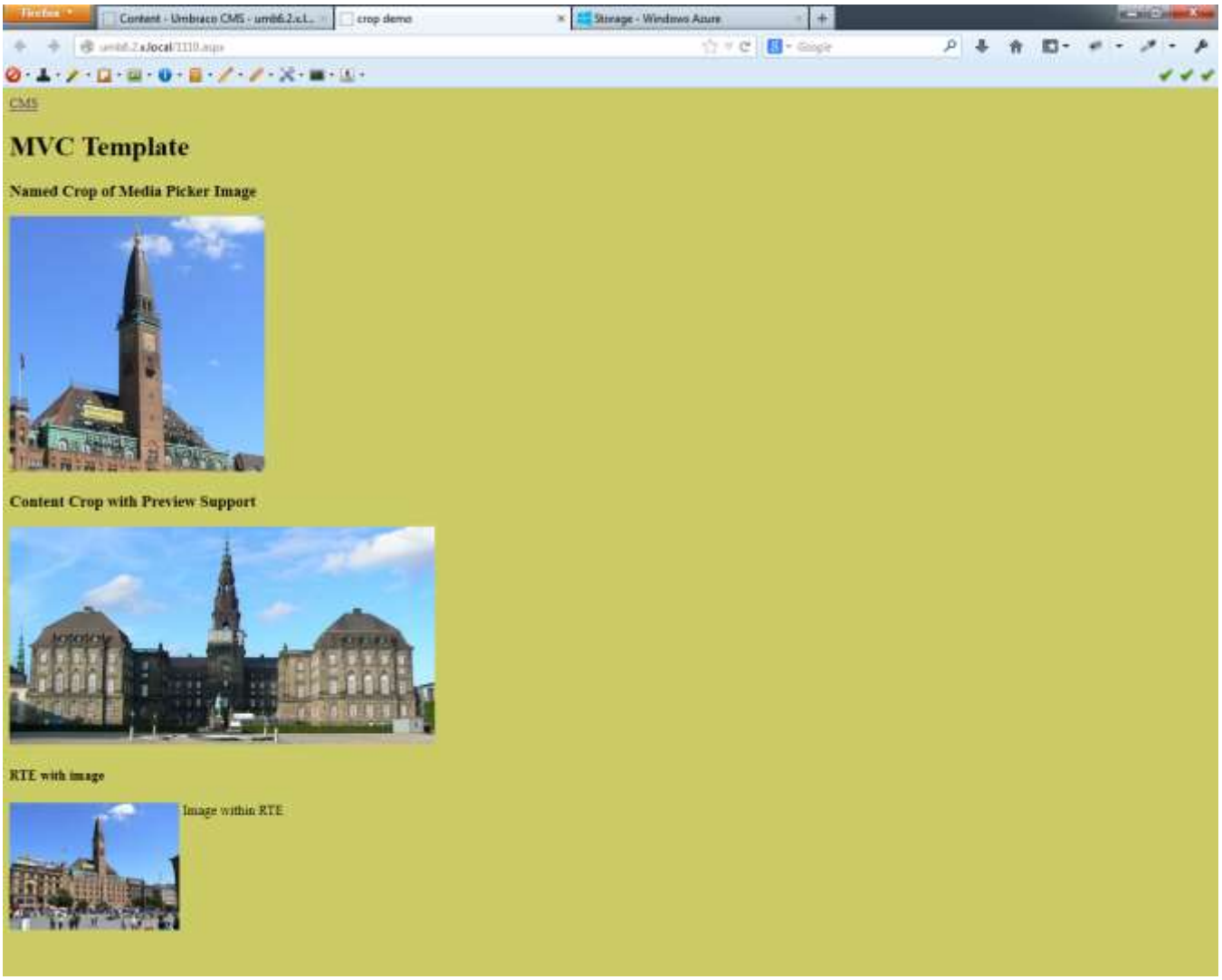
10. Sample scripts for a partial view ...



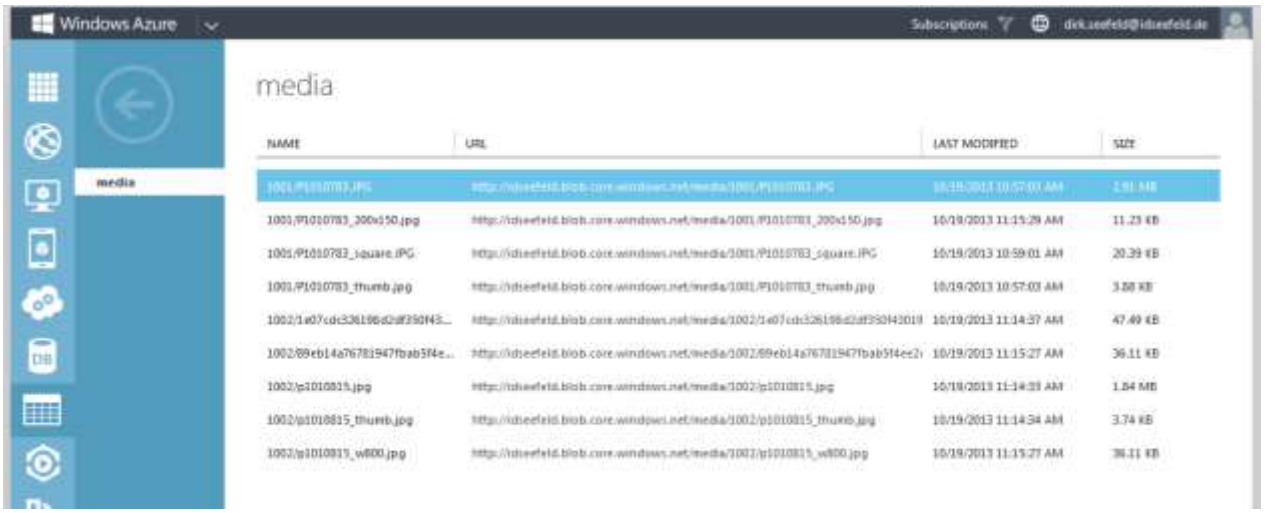
... and one for the MVC template, available in the documentation section of the [package download page](#).



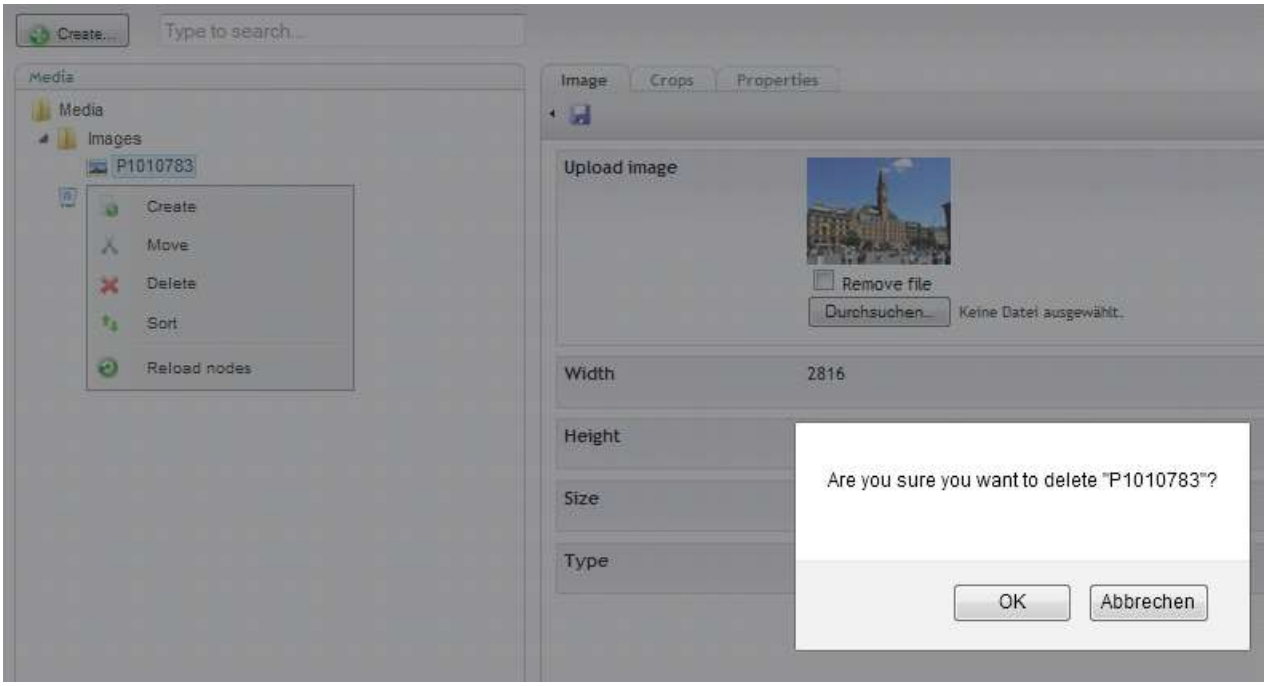
11. Result view of the sample template.

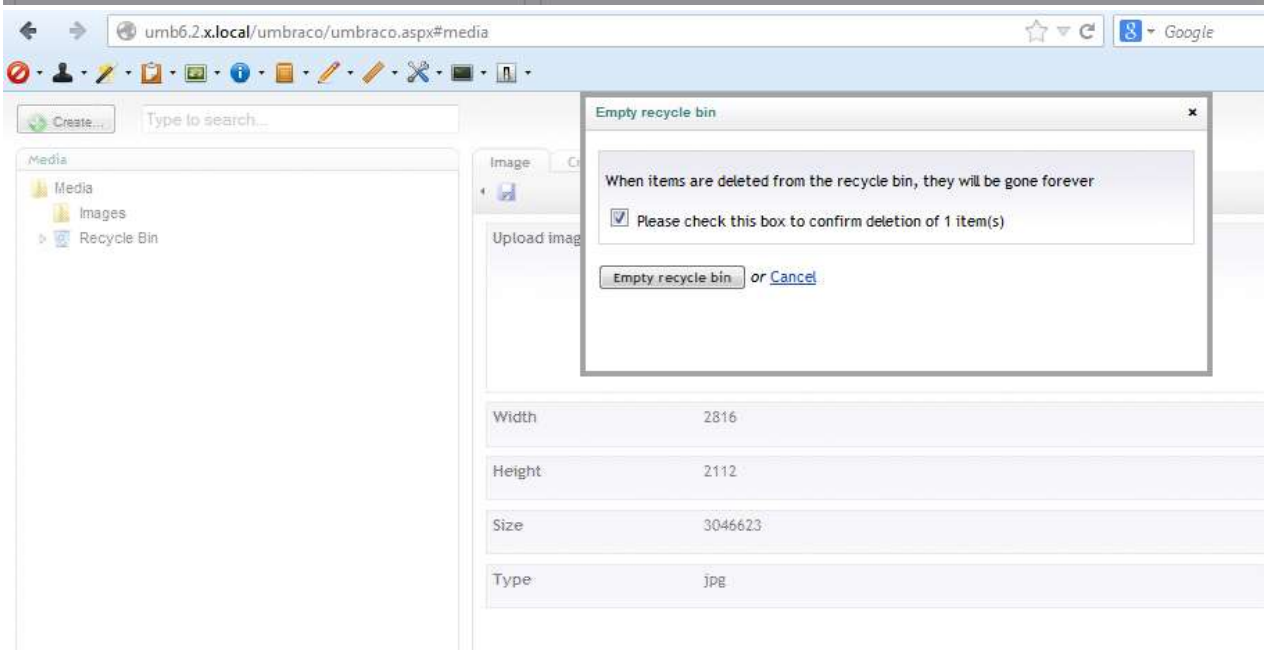


12. List of uploaded files.

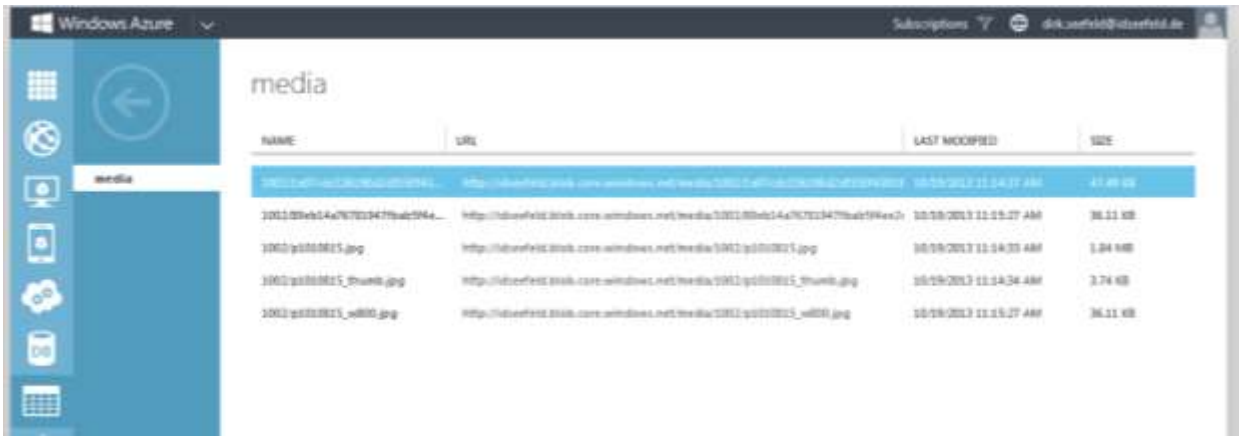


13. Delete a media item.





At the time of this writing (previous to version 6.2.0) there is an issue with deleting a single media item in the recycle bin.
But you can empty the whole recycle bin and all trashed items will be deleted in Azure storage.



An [issue report] with a possible solution is created, but I don't know whether it will make it into the release version 6.2.0 - your vote might help.

14. Bug Fix
    Until version 1.0.5 there was a bug that lead to doubled directory names (actually numbers starting with 1001) for different uploads. In case of deleting an item this actually deletes the whole directory, as this is assumed unique for only one item and siblings like thumbnails for images.
    Version 1.0.8 writes a control file to the repository with the number of the latest existing media directory in case you have data from a previous version. Otherwise, it contains 0. This prevents “old” incorrectly named blobs from deletion. Of course, this is only a work-a-round. However, it is the best solution I could figure out so far.

Localhost with local Domain Redirecting to Windows Azure Storage Emulator

For development purpose you want to use a local emulator instead of real Storage in the Azure Cloud. You get such an emulator with the **Microsoft Azure SDK** from Microsoft.
By default the blob storage of the *WAStorageEmulator* runs on a localhost, port 10000:

http://127.0.0.1:10000

an image URL would look like:

http://127.0.0.1:10000/devstoreaccount1/media/1016/p1010814.jpg

This leads to an issue with Umbraco's ImageProcessor, which ie. resizes crops (and a lot of other image manipulations) on the fly. For *external* resources like the Azure Blob Storage (cloud or emulator) you have to use the remote.axd feature and an image url would look like:

http://umbraco7nuget/remote.axd/127.0.0.1:10000/devstoreaccount1/media/1016/p1010814.jpg?anchor=center&mode=crop&width=900&height=400&rnd=130744622710000000

The default or recommended configuration of IIS will not allow colons in a URL segment like 127.0.0.1:10000. There are httpRuntime parameters that will make colons possible… However these are strictly NOT recommended. Read Scott Hanselman blog about this topic.

One simple solution would be to create a website in your local IIS with a locally defined domain like azure-emu with a simple HTTP redirect to http://127.0.0.1:10000. Then you can rewrite the above image URL to:

http://umbraco7nuget/remote.axd/azure-emu/devstoreaccount1/media/1016/p1010814.jpg?anchor=center&mode=crop&width=900&height=400&rnd=130744622710000000

Issue solved!

For example you can get such an image URL this way:

```
var picture = Umbraco.TypedMedia(page.GetPropertyValue<int>("testpicture"));
string cropUrl = picture == null ? ""
    : picture.GetCropUrl("landscape")
        .Replace("127.0.0.1:10000", "azure-emu")
        .Replace("://", "://" + Request.Url.DnsSafeHost + "/remote.axd/");
```

Related links:
http://imageprocessor.org/imageprocessor-web/
http://imageprocessor.org/imageprocessor-web/configuration/
http://imageprocessor.org/imageprocessor-web/plugins/azure-blob-cache/

http://www.hanselman.com/blog/ExperimentsInWackinessAllowingPercentsAnglebracketsAndOtherNaughtyThingsInTheASPNETIISRequestURL.aspx